

# Optimizing Index based on Text Clustering by Graph based Versions of KNN and AHC Algorithm

Taeho Jo  
President  
Alpha AI Research  
Cheongju, South Korea  
tjo018@naver.com

**Abstract**—In this research, we propose the graph based KNN algorithm and the graph based AHC algorithm for reinforcing the index optimization automation, using the text clustering. In the previous work, even if the graph based KNN version was proposed as an approach to the index optimization, it was required to present a domain as an input text tag for performing the index optimization. In this research, text clusters are constructed based on their contents by the text clustering, and a novice text index are optimized by selecting a classifier which corresponds to its most similar cluster. The AHC algorithm which clusters graphs and the KNN algorithm which classifies a graph are adopted respectively as the approaches to the text clustering and the index optimization. The goals of this research are to automate the keyword extraction and to improve the performance of both tasks, using the graph-based algorithms.

## I. INTRODUCTION

The index optimization is the process of optimizing the index by the index expansion and the index filtering, to maximize the information retrieval performance. It is necessary to define the important classes in advance, to decide whether to apply the index expansion or the index filtering to each word. The three important classes, expansion, inclusion, and removal are defined, and the index optimization is interpreted into a classification task where each word is classified into one of the three classes. The index expansion is applied to words which are classified into expansion, nothing is applied to ones which are classified into inclusion, and the index filtering is applied to ones which are classified into removal. This research focuses on the classification task, to provide the better approach, rather than to implement the index optimization system.

The motivation of this research is the requirement of deciding domains for designing the index optimization system. The index optimization is viewed into a classification task, to apply a machine learning algorithm; the index optimization system is decomposed into independent classification systems as many as domains. The design of the index optimization system is to predefine domains depending on prior knowledge, prepare sample examples domain by domain, and allocate a classifier for each domain. One more motivation for this research is the successful results from applying the graph based KNN algorithm and the graph

based AHC algorithm, respectively, to the index optimization and the text clustering. This research is intended to automate the domain decision by connecting the index optimization with the text clustering.

The idea of this research is to connect the index optimization with the text clustering and apply the graph based KNN and the graph based AHC algorithm to both tasks. In this research, we prepare the text collection where each text is associated with words which are labeled with one of expansion, inclusion, and removal. The similarity metric between two graphs based on the similarity metric between two edges, and the KNN algorithm and the AHC algorithm are modified into the graph-based versions as the approaches to both tasks. The collection is segmented into clusters by the text clustering, and each cluster is interpreted as a domain to which a classifier is allocated. A domain is automatically decided to an input text by its similarities with the cluster prototypes for nominating a corresponding classifier.

Let us mention some benefits from this research. The main problems in encoding words and texts into numerical vectors are avoided by encoding them into graphs. The performance is improved by adopting the graph-based versions of the KNN algorithm and the AHC algorithm as the approaches to the index optimization and the text clustering. We do not need prior knowledge and subjective views for defining domains by automating it through the text clustering. It does not require to present a domain by automating the domain decision based on the similarities of an input text with the cluster prototypes.

Let us mention some remaining tasks as further discussions on this research. More advanced machine learning algorithms and deep learning algorithms are modified into the graph-based versions. It is necessary to define and characterize mathematically more operations on graphs. The index optimization system which relates with the text clustering should be implemented as a real program or a library. The text classification or the text clustering may relate to this system for improving their performances.

## II. PROPOSED APPROACH

### A. Similarity Metric

This section is concerned with the graph as a word representation and the similarity between graphs. In representing a word into a graph, a vertex indicates a text, and an edge indicates a similarity between texts. The similarity between edges is computed, before computing the similarity between graphs, viewing a graph as a set of edges. The similarity between graphs is computed by averaging over the similarities of all possible pairs of edges. In this section, we describe the process of encoding a word into a graph and the process of computing the similarity between graphs.

The process of encoding words into graphs is illustrated in Figure 1. In the vertex definition, from a text collection, texts which are relevant to the word are retrieved as the vertices. In the edge definition, the similarities among the retrieved texts are computed as edges. In the edge selection, the edges with their higher similarities are selected for representing a graph. Refer to [2] for studying the encoding process in more detail.

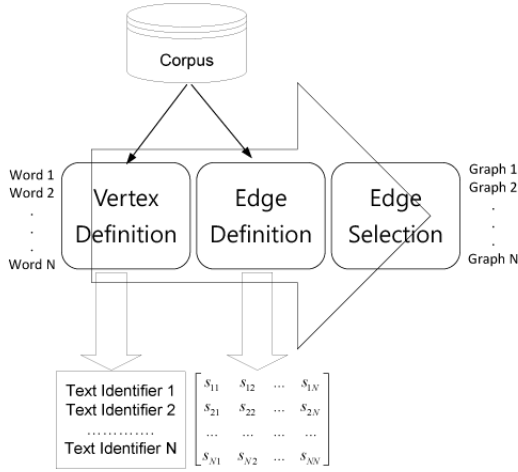


Figure 1. Process of Encoding Words into Graphs

The three cases of computing the similarity between two edges are illustrated in Figure 2. Each weight which is associated with its own edge is assumed to be a normalized value between zero and one, and if both nodes are identical to each other, the average over two weights is the similarity between two edges. If either of the two nodes is identical to each other, the product of two weights is the similarity between two edges. If neither of two nodes is identical, zero is the similarity between them. The similarity between two edges is always a normalized value between zero and one.

Let us mention the process of computing the similarity between graphs as a semantic similarity between words. The two graphs which represent words are notated by edge sets,  $G_1 = \{e_{11}, e_{12}, \dots, e_{1|G_1|}\}$  and  $G_2 = \{e_{21}, e_{22}, \dots, e_{2|G_2|}\}$ . Two edges,  $e_{1i} \in G_1$  and  $e_{2j} \in G_2$ ,

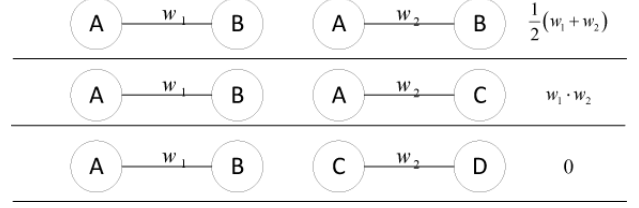


Figure 2. Three Cases of Comparing Two Edges with Each Other

are associated with their own weights,  $w_{1i}$  and  $w_{2j}$ , and the similarity between two edges,  $e_{1i}$  and  $e_{2j}$  by equation (1), (2), or (3), depending on the cases which are mentioned above,

$$\text{sim}(e_{1i}, e_{2j}) = \frac{1}{2}(w_{1i} + w_{2j}) \quad (1)$$

$$\text{sim}(e_{1i}, e_{2j}) = w_{1i} \times w_{2j} \quad (2)$$

$$\text{sim}(e_{1i}, e_{2j}) = 0 \quad (3)$$

The similarity between two graphs is computed by equation (4),

$$\text{sim}(G_1, G_2) = \frac{1}{|G_1| \times |G_2|} \sum_{i=1}^{|G_1|} \sum_{j=1}^{|G_2|} \text{sim}(e_{1i}, e_{2j}). \quad (4)$$

The value which is generated from equation (4), is always given as a normalized value between zero and one as shown in equation (5),

$$0 \leq \text{sim}(G_1, G_2) \leq 1. \quad (5)$$

Let us make some remarks on the process of encoding a word into a graph and computing the similarity between graphs. A word is encoded into a graph by the vertex definition, the edge definition, and the edge weighting. The three cases are considered for computing the similarity between edges. The similarity between graphs is computed by equation (4). In the future research, we consider representing a word into multiple graphs.

### B. Graph based KNN Algorithm

This section is concerned with the graph based KNN algorithm. It was initially proposed as the approach to the word classification by Jo in 2018 [2]. The similarity metric between graphs which was described in the previous section is used for computing the similarity between a novice graph and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into graphs, and they are notated by a set,  $Tr =$

$\{G_1, G_2, \dots, G_N\}$ . A novice word is encoded into a graph notated by  $G$ . Its similarities with the graphs which represent the sample words are computed by equation (4), as  $sim(G, G_1), sim(G, G_2), \dots, sim(G, G_N)$ . Some training examples which are most similar as the novice input are selected as its nearest neighbors.

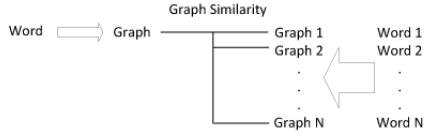


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (6),

$$Nr = Select_k(Tr, G), Nr \subseteq Tr \quad (6)$$

The set of nearest neighbors,  $Nr$ , is composed with elements as expressed in equation (7),

$$Nr = \{G_1^{near}, G_2^{near}, \dots, G_k^{near}\} \quad (7)$$

The elements in the set,  $Nr$ , are used for voting their labels in the next step.

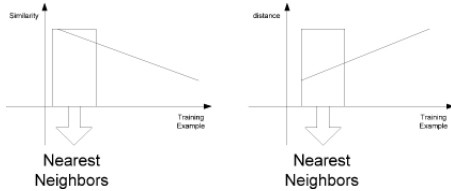


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by  $c_1, c_2, \dots, c_m$ , and the label of a nearest neighbor is notated by  $c_j = label(G_i^{near})$ . The number of nearest neighbors which belong to the category,  $c_j$ , is notated by  $Count(Nr, c_j)$ . The label of the novice item,  $G$ , is decided by equation (8),

$$label(G) = \arg\max_{j=1}^m Count(Nr, c_j) \quad (8)$$

The process of deciding the label of a novice item by equation (8), is called voting.

Let us make some remarks on the graph based KNN algorithm. It computes the similarities of a novice item with the training examples. The training examples with their

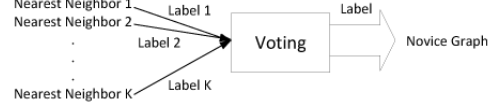


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

### C. System Architecture

This section is concerned with the index optimization system which adopts the graphs based KNN algorithm. In Section II-B, we described the proposed version of KNN algorithm as the approach to the index optimization. It is viewed as a classification task which classifies a word into one of the three categories. This section is intended to describe the index optimization system with respect to its function and its architecture.

The process of collecting the same words and encoding them into graphs for implementing the index optimization system is illustrated in Figure 6. The index optimization is interpreted into a domain dependent classification; even a same word is classified differently, depending on the domain. For each domain, an independent classification task is defined. The several domains are decided, and the words which are labeled with one of the three categories exclusively in each domain. It is required to present the domain from a text which is given as the input for doing this task.

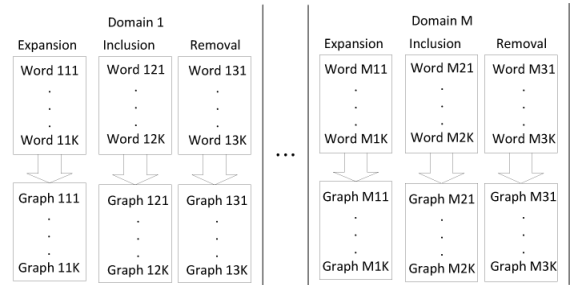


Figure 6. Preparation of Training Examples

The entire architecture of the proposed index optimization system is illustrated in Figure 7. A text is given as the input, and words are extracted from it in the indexing module. In the encoding module, the sample words in the groups, expansion, inclusion, and removal and ones which are indexed from the text are encoded into graphs. In the similarity computation module and the voting module, the words which are indexed from the text are classified into

one of the three categories. The words which are classified into removal are discarded in this system.

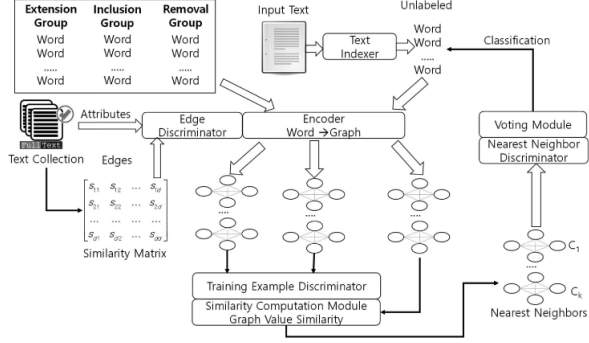


Figure 7. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 8. Sample words which are labeled with one of the three categories are collected within a domain, and they are encoded into graphs. An input text is indexed into a list of words, and they are also encoded into graphs. The nearest neighbors are selected by the similarity computation, and the label is decided by voting ones of their nearest neighbors for each word. Ones which are classified into expansion require their associated words from external sources, ones which are classified into inclusion are preserved, and ones which are classified into removal are excluded.

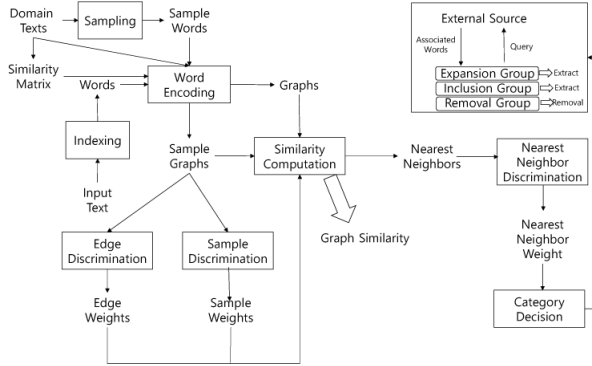


Figure 8. System Flow

Let us make some remarks on the proposed system which is illustrated in Figure 7 as its architecture. The index optimization is defined as a multiple classification task where each word is classified into expansion, inclusion, or removal. The words are encoded into graphs instead of numerical vectors, and they are classified directly. The words which are classified into removal are excluded from index of the input text. In implementing the system as its complete version, we need to add the module which retrieve more words which are relevant to ones which are labeled with expansion.

#### D. Connection with Text Clustering

This section is concerned with the connection of the index optimization with the text clustering. In the previous section, we mentioned the index optimization as an instance of domain dependent classification. The domains are automatically constructed from the text collection by clustering texts. The AHC algorithm which is proposed in [3] is used for implementing the text clustering. This section is intended to describe the index optimization system which is connected with the text clustering for automating the construction of domains.

The architecture of the text clustering system which was proposed in [3] is illustrated in Figure 9. The texts in the group are encoded into graphs, and the AHC algorithm which is proposed in [3] is adopted as the approach. It starts with singletons as many as items, computes the similarities of all possible cluster pairs, and merge the cluster pair with its highest similarity into one cluster. The two steps are iterated until the number of clusters reach the desired number. We may consider replacing the AHC algorithm by its variants for improving the speed and the performance.

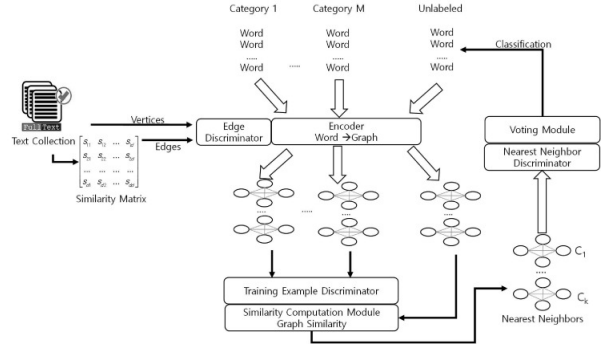


Figure 9. Text Clustering System

The connection of the index optimization with the text clustering is illustrated in Figure 10. The  $M$  domains are defined by clustering texts in a group, initially and automatically. A novice text is given as the input, and its domain, domain  $D$ , is decided by its similarities with domains. A classifier which corresponds to domain  $D$  is nominated and classify each word in a novice text into expansion, inclusion, or removal. The  $M$  index optimization systems are viewed as independent ones, each of which classifies each word into one of the three categories.

The execution flow of index optimization which is connected with the text clustering is illustrated in Figure 11. Unlabeled texts are clustered into subgroups, each of which is a domain. Sample words which are labeled with expansion, inclusion, or removal are generated from each domain. These sample words are provided for training the classifier in each index optimization system. Each classifier is used for judging the importance level of each word which is indexed

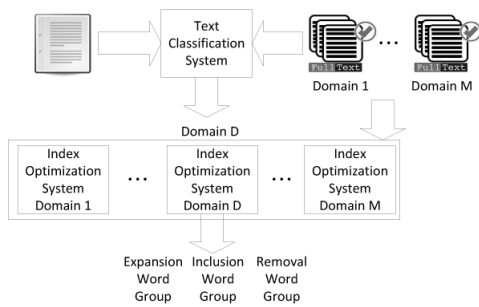


Figure 10. Index Optimization System connected with Text Clustering

from the input text.

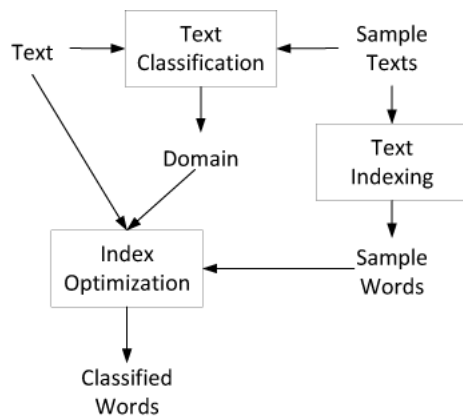


Figure 11. System Flow of Index Optimization connected with Text Clustering

Let us make some remarks on the connection of the index optimization with the text clustering. It is the process of segmenting a text group into subgroups based on their content-based similarities. The role of text clustering is to define the domains initially in the architecture of connecting the index optimization with the text clustering. In each domain, sample words are generated, and its corresponding classifier is trained with them. In the future research, we consider using the index optimization for clustering texts in the opposite direction.

## REFERENCES

- [1] T. Jo, "Graph based KNN for Optimizing Index of News Articles", 53-62, Journal of Multimedia Information System, 3, 2016.
- [2] T. Jo, "Comparing Graph based K Nearest Neighbor with Traditional Version in Word Categorization in NewsPage.com, 12-18, International Journal of Advanced Social Sciences, 1, 2018.
- [3] T. Jo, "Graph based Version for Clustering Texts in Current Affair Domain", 171-176, The Proceedings of 15st International Conference on Data Science, 2019.